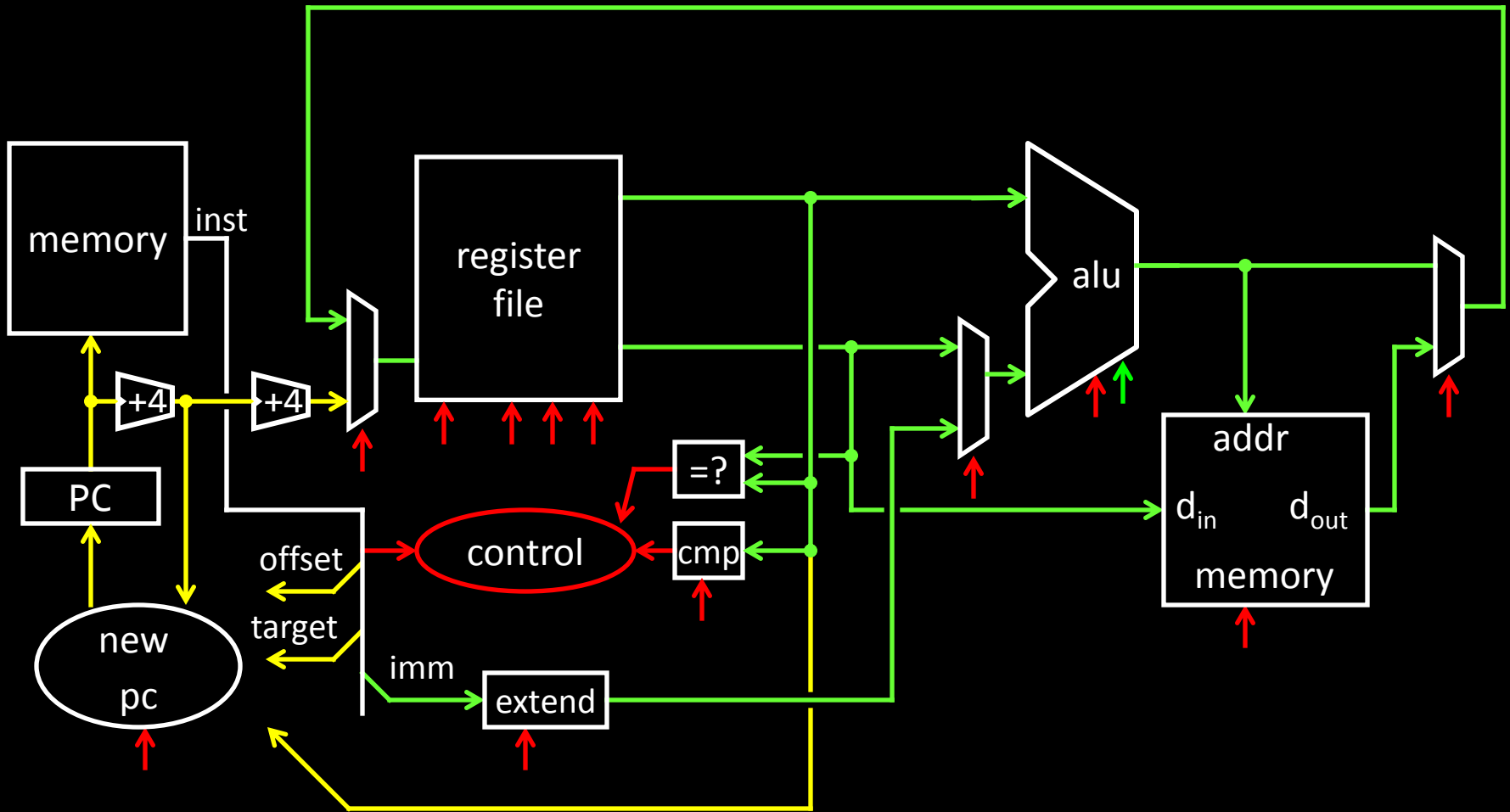

Memory

Hakim Weatherspoon
CS 3410, Spring 2012
Computer Science
Cornell University

See: P&H Appendix C.8, C.9

Big Picture: Building a Processor



A Single cycle processor

Goals for today

Review

- Finite State Machines

Memory

- Register Files
- Tri-state devices
- SRAM (Static RAM—random access memory)
- DRAM (Dynamic RAM)

Example: Digital Door Lock



Digital Door Lock

Inputs:

- keycodes from keypad
- clock

Outputs:

- “unlock” signal
- display how many keys pressed so far

Door Lock: Inputs

Assumptions:

- signals are synchronized to clock
- Password is B-A-B



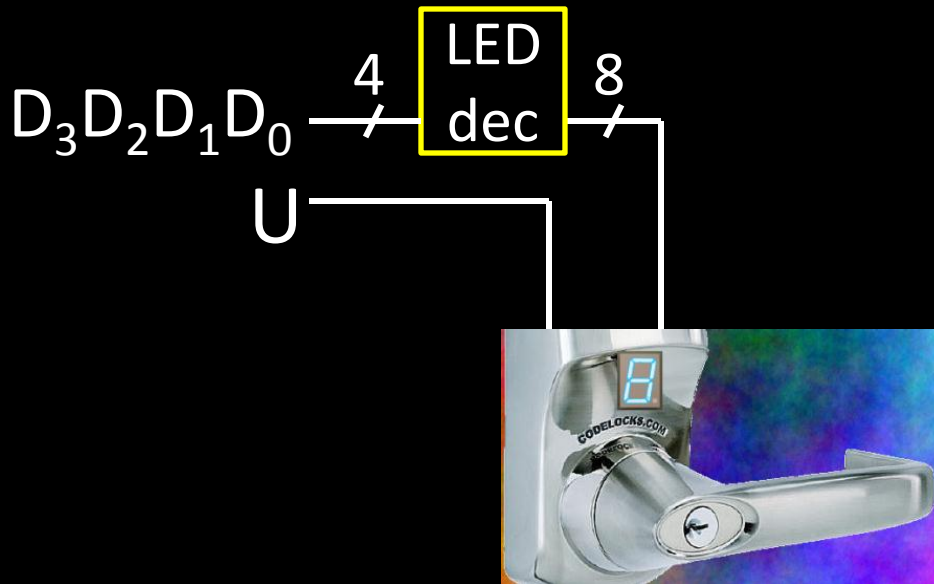
K
A
B

K	A	B	Meaning
0	0	0	\emptyset (no key)
1	1	0	'A' pressed
1	0	1	'B' pressed

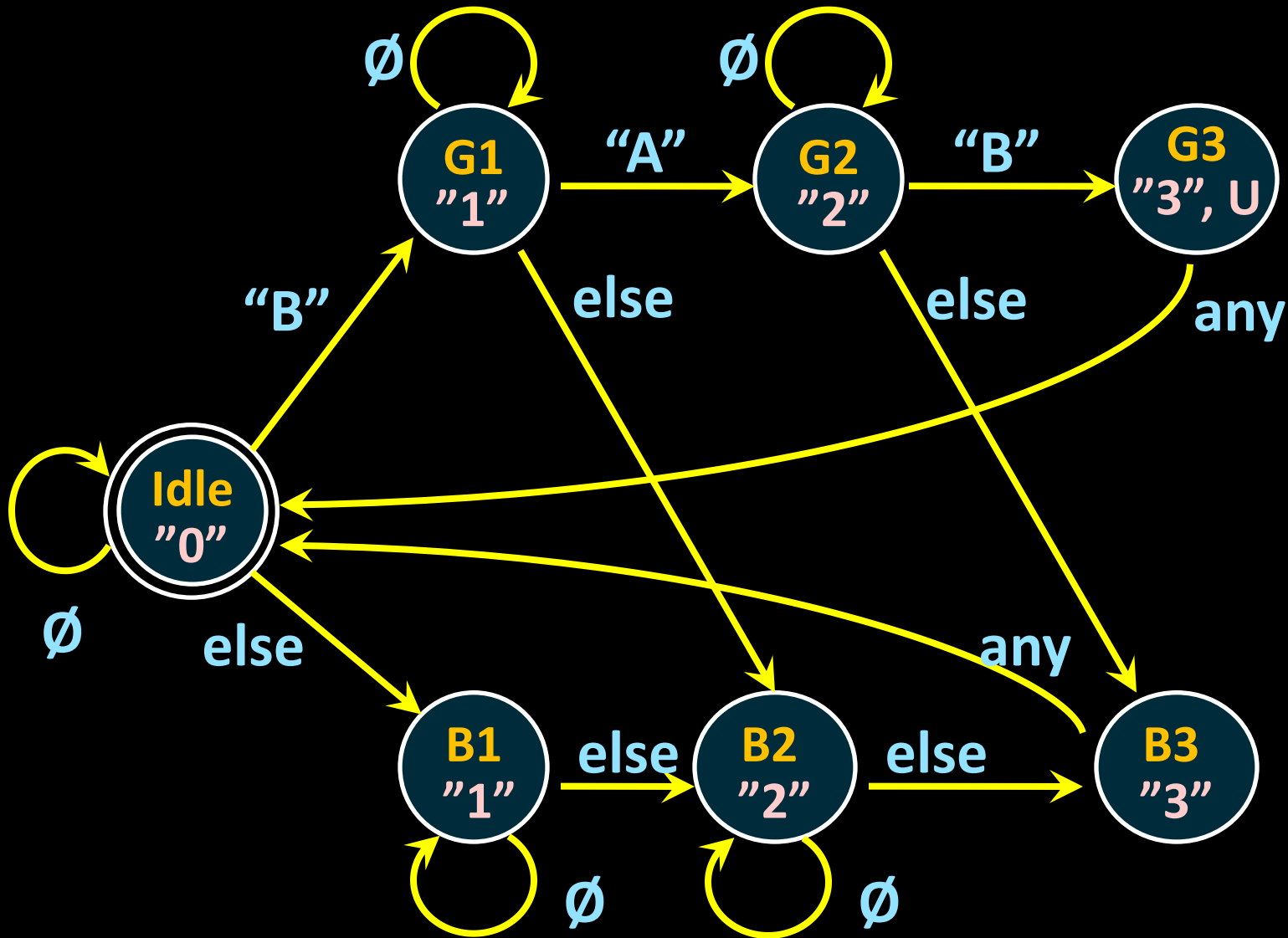
Door Lock: Outputs

Assumptions:

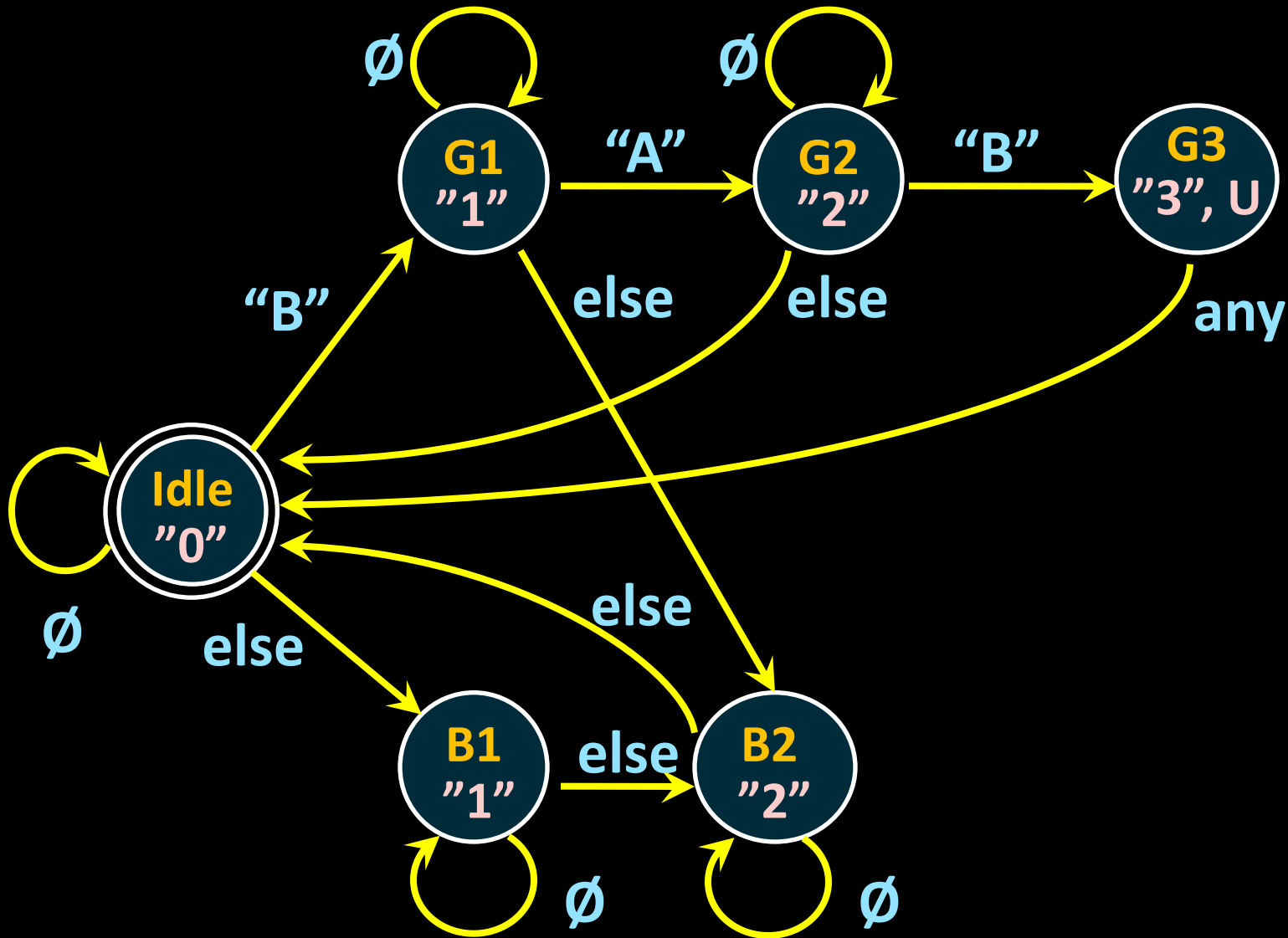
- High pulse on U unlocks door



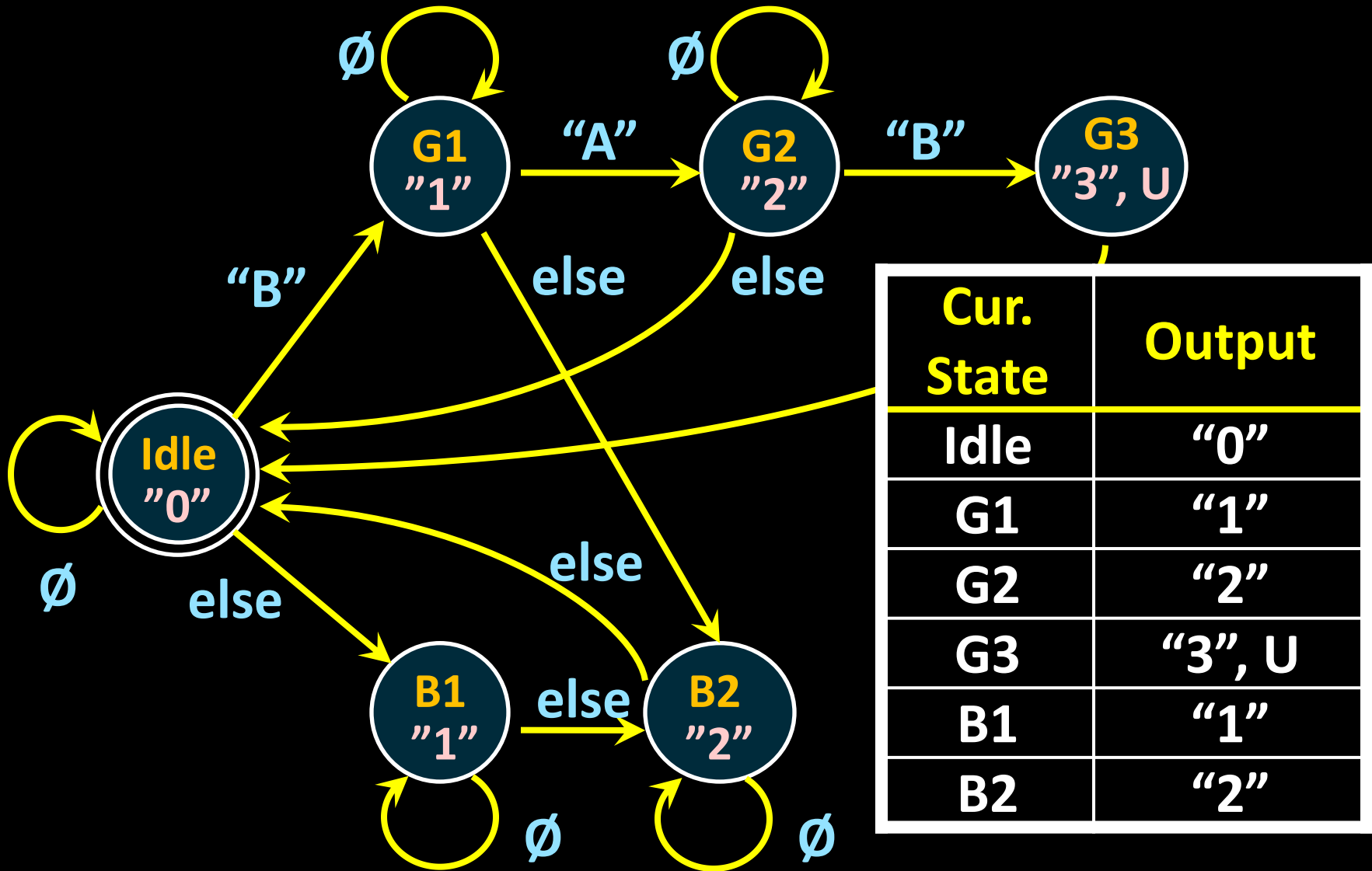
Door Lock: Simplified State Diagram



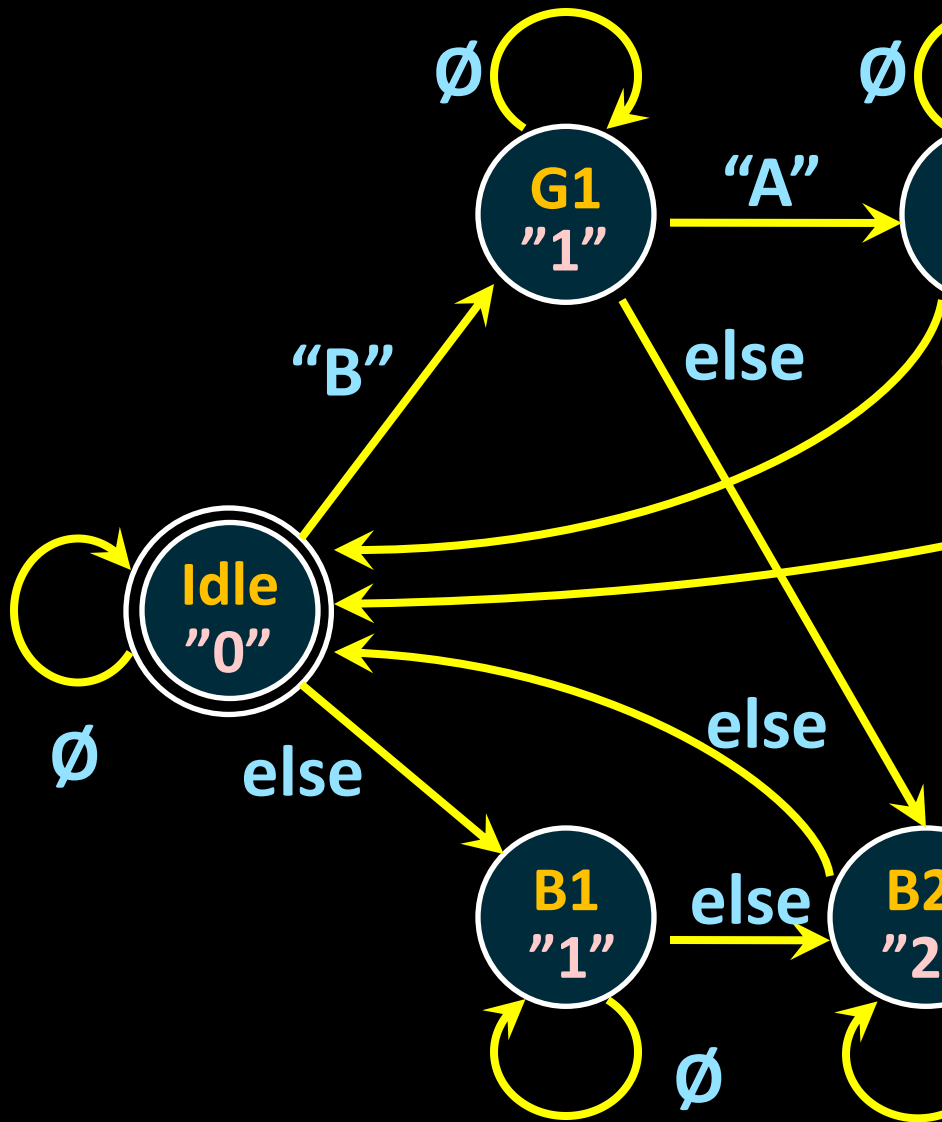
Door Lock: Simplified State Diagram



Door Lock: Simplified State Diagram



Door Lock: Simplified State Diagram



Cur. State	Input	Next State
Idle	\emptyset	Idle
Idle	"B"	G1
Idle	"A"	B1
G1	\emptyset	G1
G1	"A"	G2
G1	"B"	B2
G2	\emptyset	B2
G2	"B"	G3
G2	"A"	Idle
G3	any	Idle
B1	\emptyset	B1
B1	K	B2
B2	\emptyset	B2
B2	K	Idle

State Table Encoding

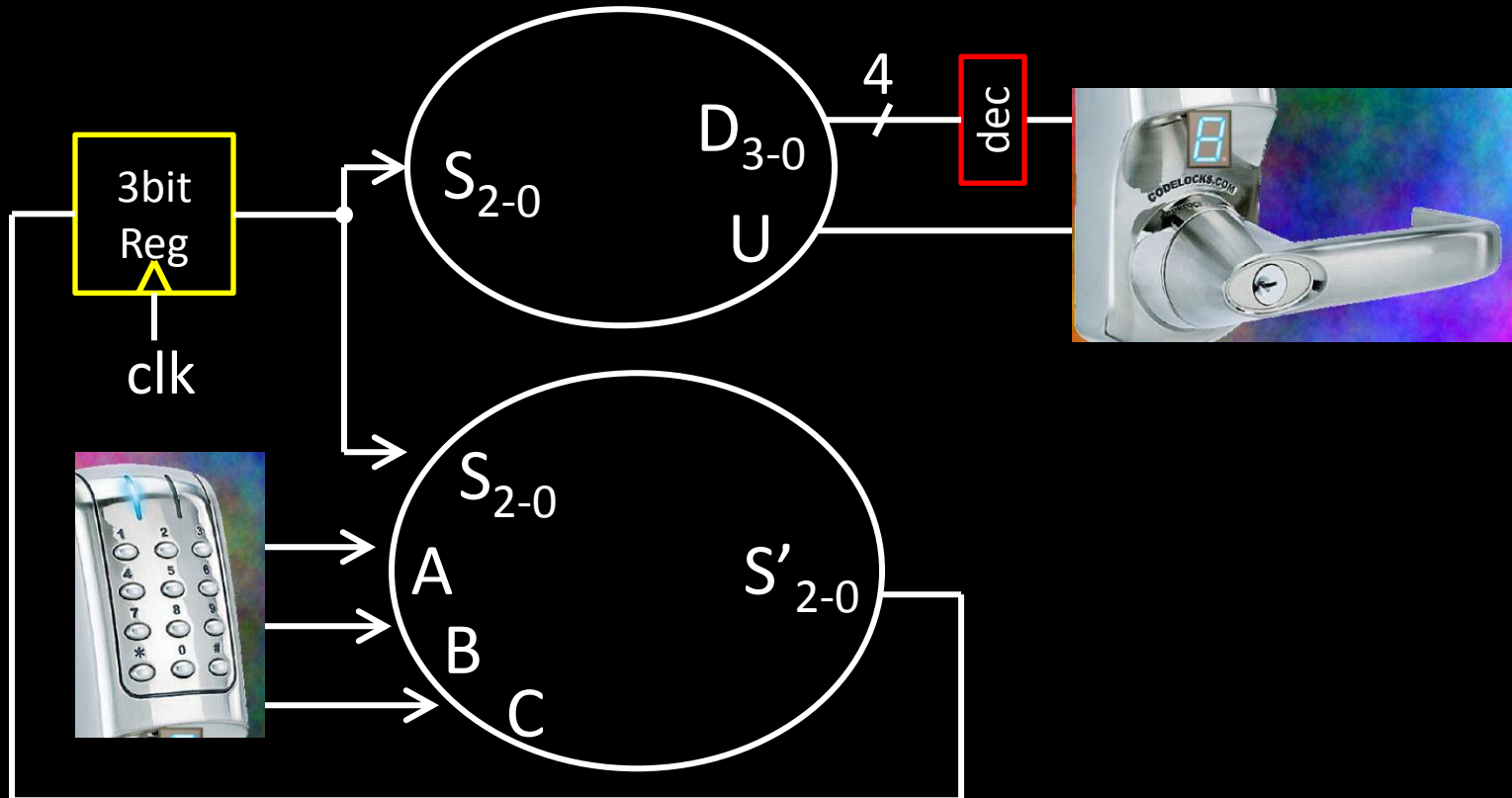
S_2	S_1	S_0	D_3	D_2	D_1	D_0	U
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	0	0	1	1	1
1	0	0	0	0	0	1	0
1	0	1	0	0	1	0	0

D_3 [

State	S_2	S_1	S_0
Idle	0	0	0
G1	0	0	1
G2	0	1	0
G3	0	1	1
B1	1	0	0
B2	1	0	1

S_2	S_1	S_0	K	A	B	S'_2	S'_1	S'_0
0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	1
0	0	0	1	1	0	1	0	0
0	0	1	0	0	0	0	0	1
0	0	1	1	1	0	0	1	0
0	0	1	1	0	1	1	0	1
0	1	0	0	0	0	0	1	0
0	1	0	1	0	1	0	1	1
0	1	0	1	1	0	0	0	0
0	1	1	x	x	x	0	0	0
1	0	0	0	0	0	1	0	0
1	0	0	1	x	x	1	0	1
1	0	1	0	0	0	1	0	1
1	0	1	1	x	x	0	0	0

Door Lock: Implementation



Strategy:

- (1) Draw a state diagram (e.g. Moore Machine)
- (2) Write output and next-state tables
- (3) Encode states, inputs, and outputs as bits
- (4) Determine logic equations for next state and outputs

Administrivia

Make sure partner in same Lab Section *this week*

Lab2 is out

Due in one week, next Monday, start early

Work **alone**

But, use your resources

- Lab Section, Piazza.com, Office Hours, Homework Help Session,
- Class notes, book, Sections, CSUGLab

No Homework this week

Administrivia

Check online syllabus/schedule

- <http://www.cs.cornell.edu/Courses/CS3410/2012sp/schedule.html>

Slides and Reading for lectures

Office Hours

Homework and Programming Assignments

Prelims (in evenings):

- Tuesday, February 28th
- Thursday, March 29th
- Thursday, April 26th

Schedule is subject to change

Collaboration, Late, Re-grading Policies

“Black Board” Collaboration Policy

- Can discuss approach together on a “black board”
- **Leave and write up solution independently**
- Do not copy solutions

Late Policy

- Each person has a **total of *four* “slip days”**
- **Max of *two* slip days** for any individual assignment
- **Slip days deducted first** for *any* late assignment, cannot selectively apply slip days
- For projects, slip days are deducted from all partners
- 20% deducted per day late after slip days are exhausted

Regrade policy

- Submit written request to lead TA,
and lead TA will pick a different grader
- Submit another written request,
lead TA will regrade directly
- Submit yet another written request for professor to regrade.

Goals for today

Review

- Finite State Machines

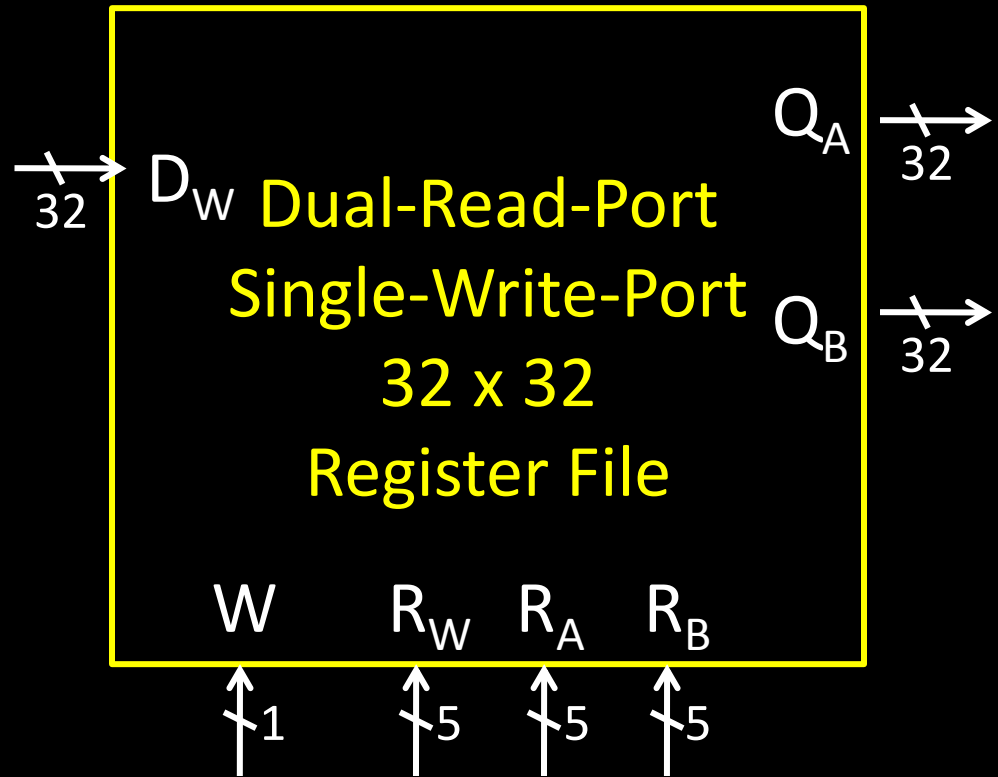
Memory

- Register Files
- Tri-state devices
- SRAM (Static RAM—random access memory)
- DRAM (Dynamic RAM)

Register File

Register File

- N read/write registers
- Indexed by register number



Implementation:

- D flip flops to store bits
- Decoder for each **write port**
- Mux for each **read port**

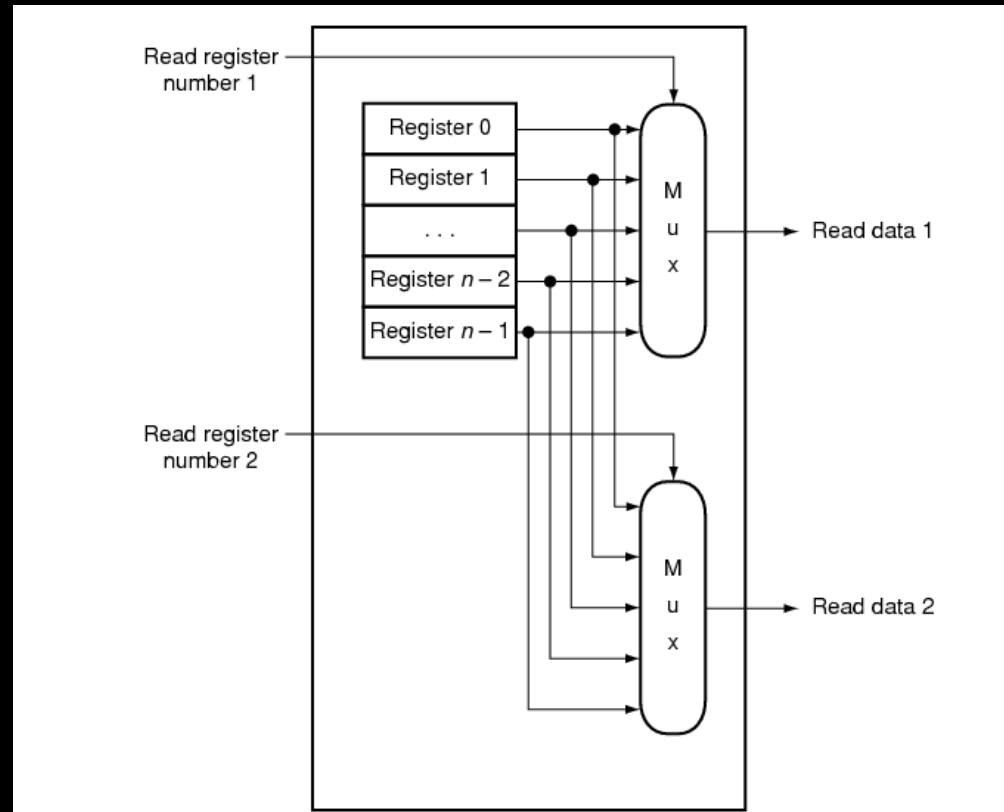
Register File

Register File

- N read/write registers
- Indexed by register number

Implementation:

- D flip flops to store bits
- Decoder for each **write port**
- Mux for each **read port**



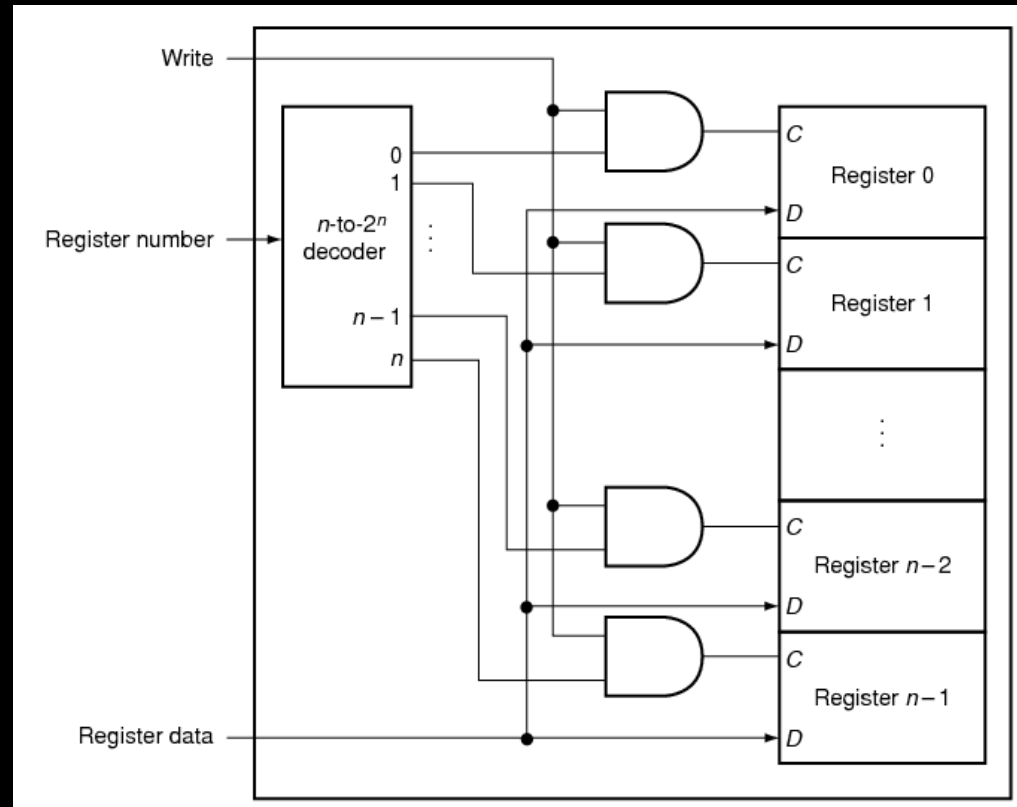
Register File

Register File

- N read/write registers
- Indexed by register number

Implementation:

- D flip flops to store bits
- Decoder for each **write port**
- Mux for each **read port**



Tradeoffs

Register File tradeoffs

- + Very fast (a few gate delays for both read and write)
- + Adding extra ports is straightforward
- Doesn't scale

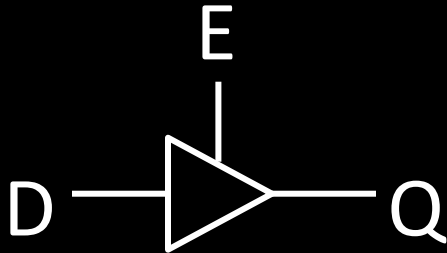
Building Large Memories

Need a shared **bus** (or shared **bit line**)

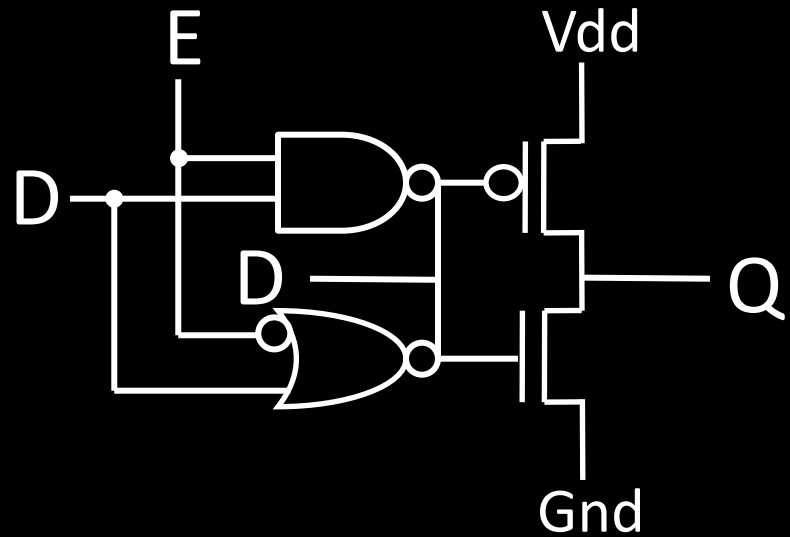
- Many FFs/outputs/etc. connected to single wire
- Only one output *drives* the bus at a time

Tri-State Devices

Tri-State Buffers

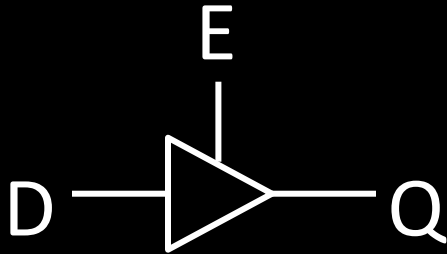


E	D	Q
0	0	z
0	1	z
1	0	0
1	1	1

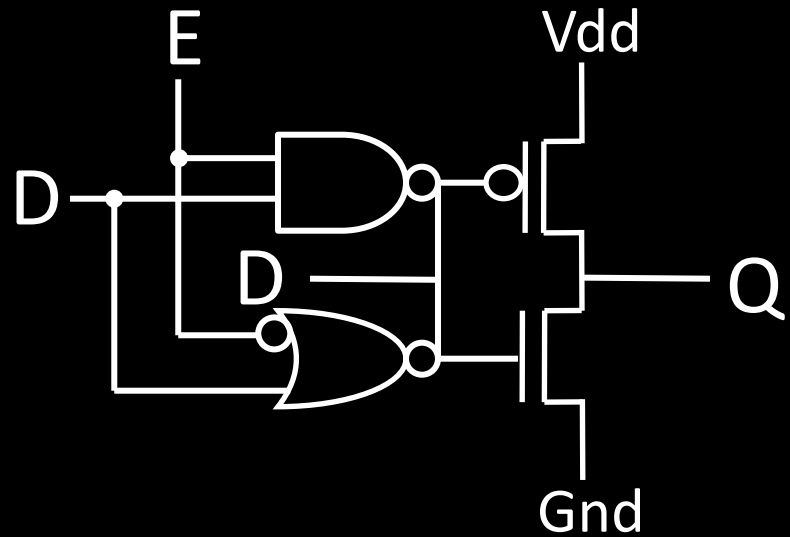


Tri-State Devices

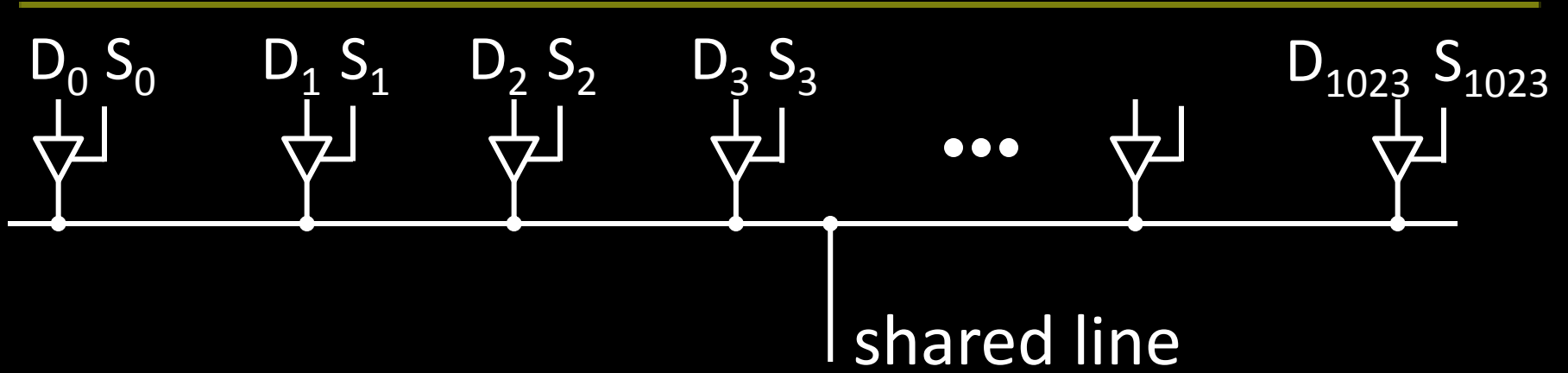
Tri-State Buffers



E	D	Q
0	0	z
0	1	z
1	0	0
1	1	1



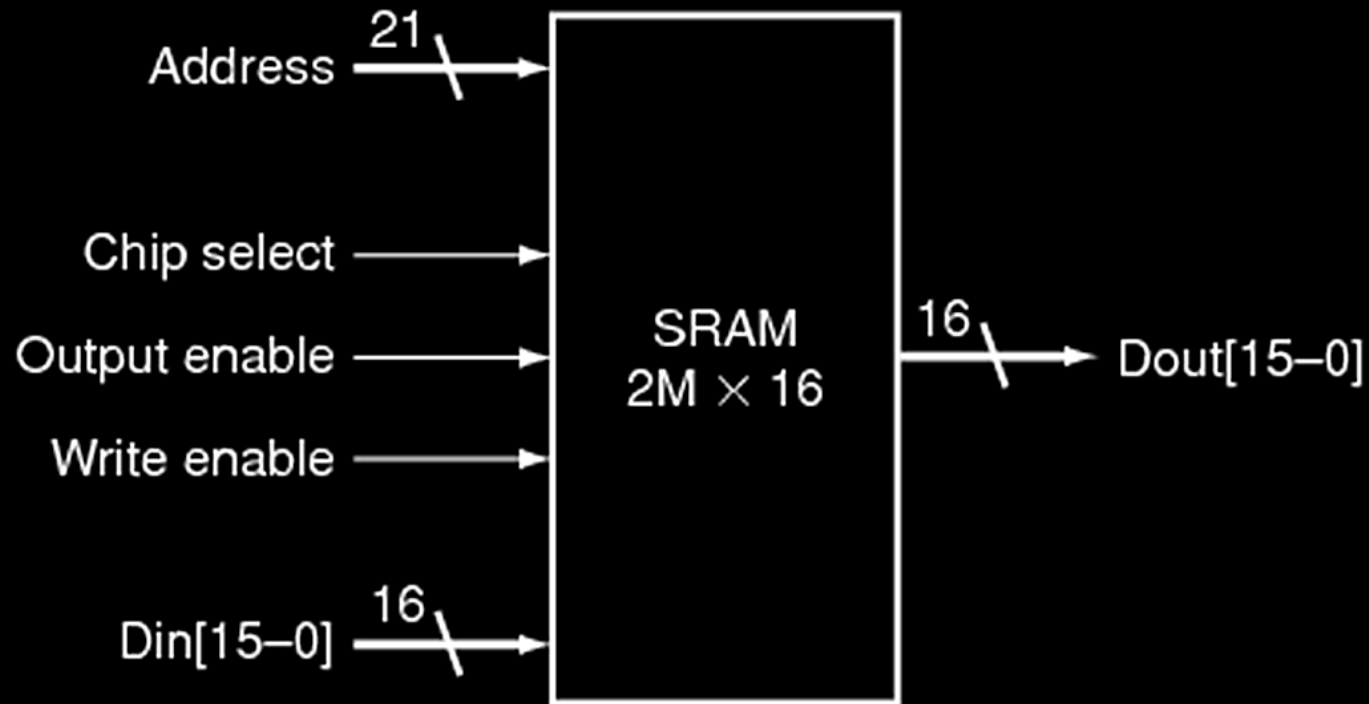
Shared Bus



SRAM

Static RAM (SRAM)

- Essentially just SR Latches + tri-states buffers



SRAM Chip

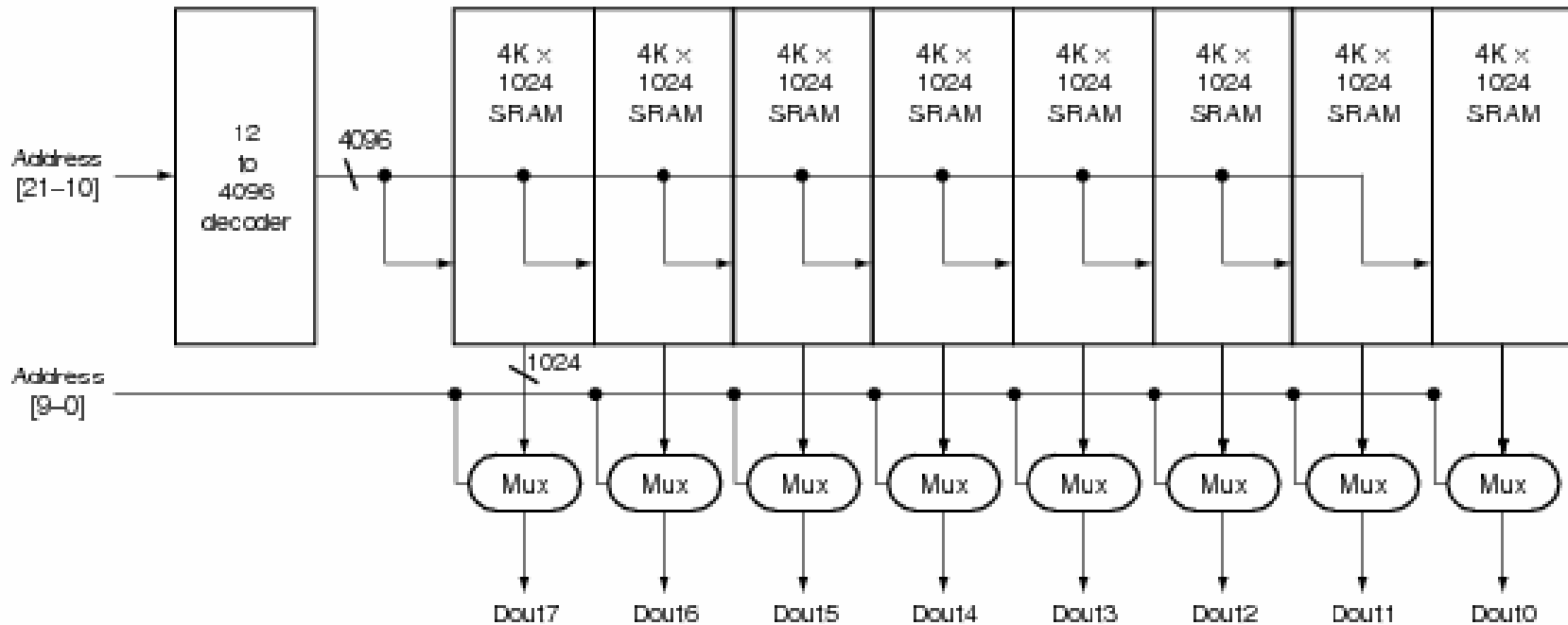
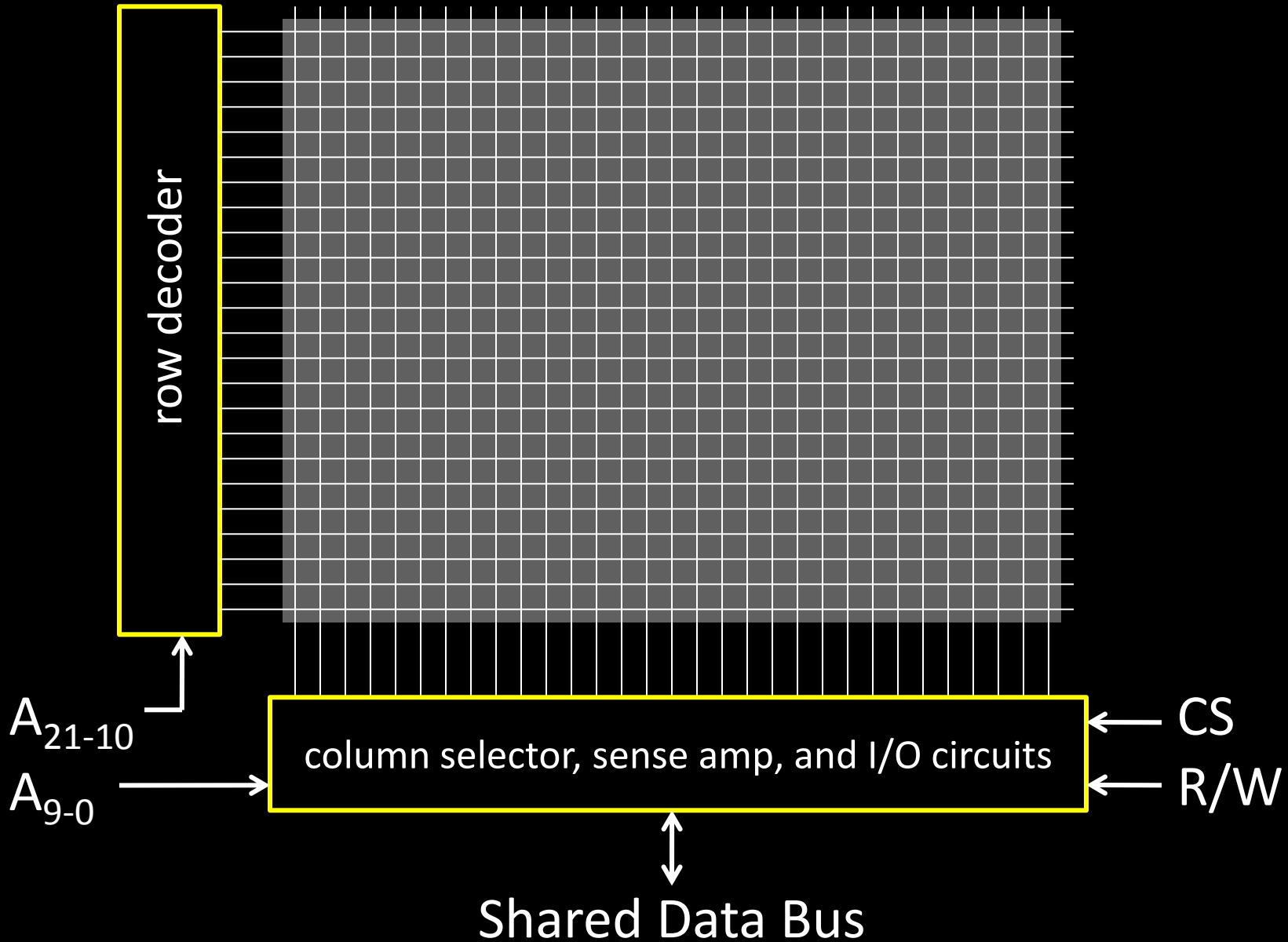


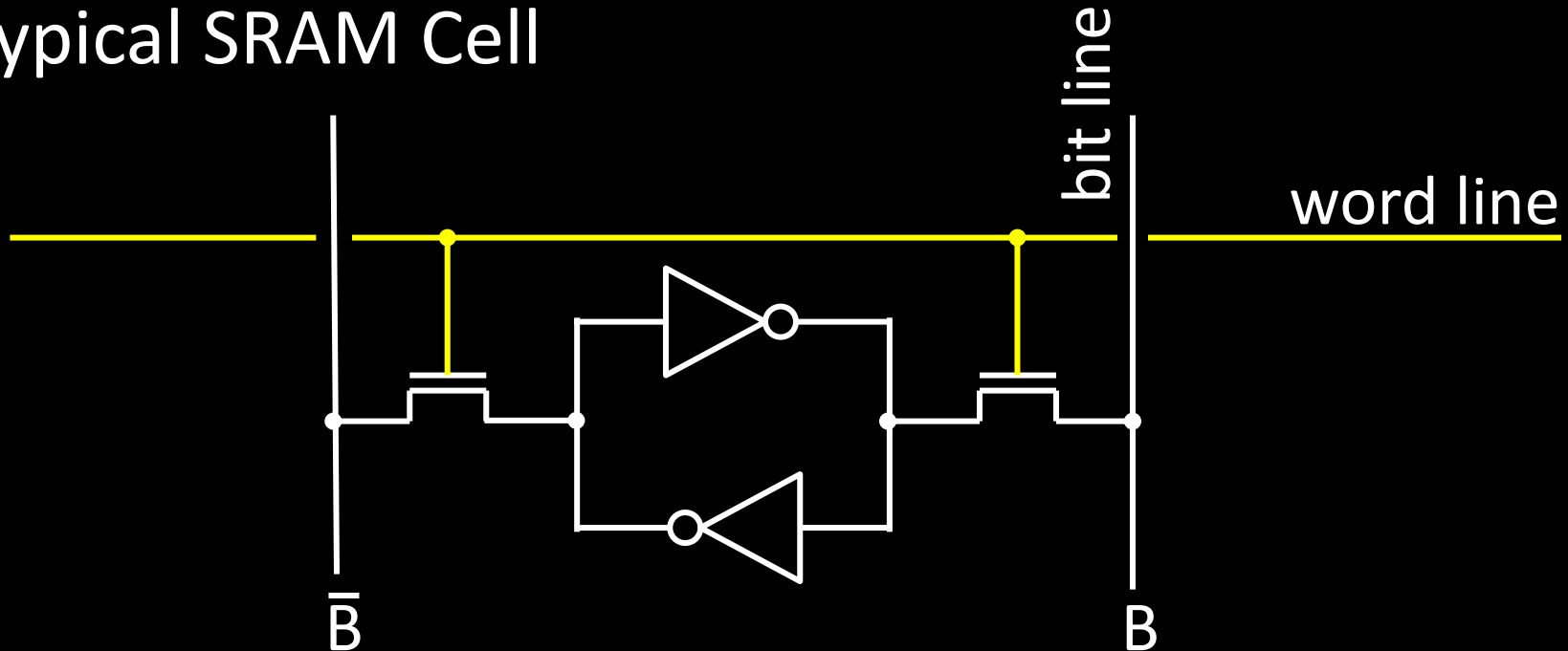
FIGURE B.9.4 Typical organization of a 4M x 8 SRAM as an array of 4K x 1024 arrays. The first decoder generates the addresses for eight 4K x 1024 arrays; then a set of multiplexers is used to select 1 bit from each 1024-bit-wide array. This is a much easier design than a single-level decode that would need either an enormous decoder or a gigantic multiplexer. In practice, a modern SRAM of this size would probably use an even larger number of blocks, each somewhat smaller.

SRAM Chip



SRAM Cell

Typical SRAM Cell



Each cell stores one bit, and requires 4 – 8 transistors (6 is typical)

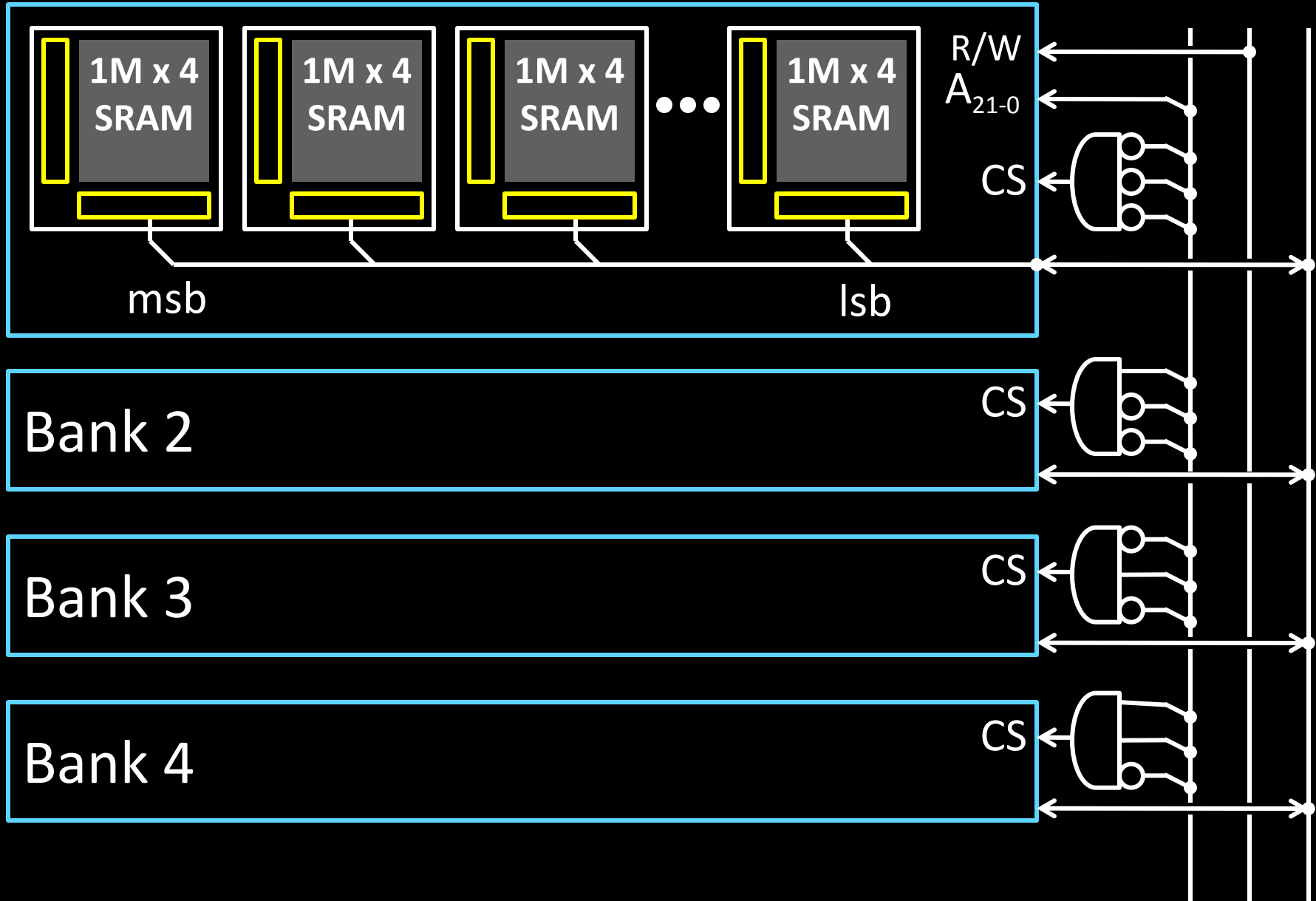
Read:

- pre-charge B and \bar{B} to $V_{dd}/2$
- pull word line high
- cell pulls B or \bar{B} low, sense amp detects voltage difference

Write:

- pull word line high
- drive B and \bar{B} to flip cell

SRAM Modules and Arrays



SRAM Summary

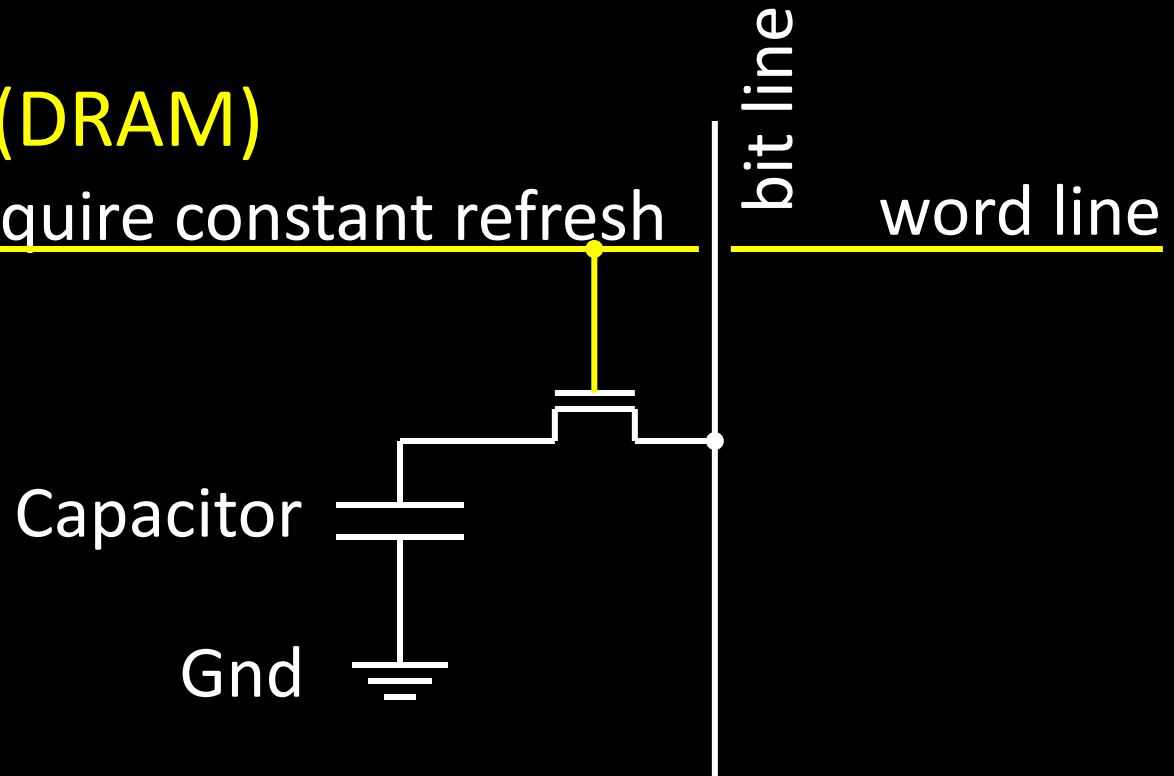
SRAM

- A few transistors (~6) per cell
- Used for working memory (caches)
- But for even higher density...

Dynamic RAM: DRAM

Dynamic-RAM (DRAM)

- Data values require constant refresh



DRAM vs. SRAM

Single transistor vs. many gates

- Denser, cheaper (\$30/1GB vs. \$30/2MB)
- But more complicated, and has analog sensing

Also needs refresh

- Read and write back...
- ...every few milliseconds
- Organized in 2D grid, so can do rows at a time
- Chip can do refresh internally

Hence... slower and energy inefficient

Memory

Register File tradeoffs

- + Very fast (a few gate delays for both read and write)
- + Adding extra ports is straightforward
- Expensive, doesn't scale
- Volatile

Volatile Memory alternatives: SRAM, DRAM, ...

- Slower
- + Cheaper, and scales well
- Volatile

Non-Volatile Memory (NV-RAM): Flash, EEPROM, ...

- + Scales well
- Limited lifetime; degrades after 100000 to 1M writes

Summary

We now have enough building blocks to build machines that can perform non-trivial computational tasks

Register File: Tens of words of working memory

SRAM: Millions of words of working memory

DRAM: Billions of words of working memory

NVRAM: long term storage

(usb fob, solid state disks, BIOS, ...)

Next time we will build a simple processor!